CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

CE143: COMPUTER CONCEPTS & PROGRAMMING

Chapter - 5

Conditional Statements & Branching

DEPSTAR

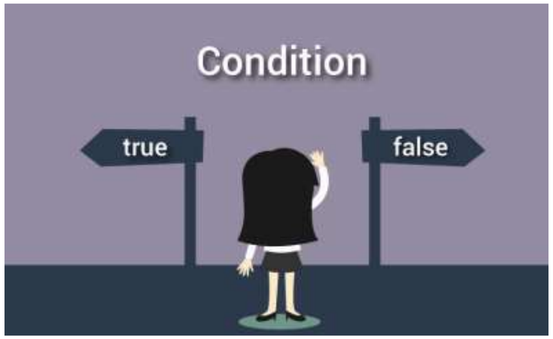
Devang Patel Institute of Advance Technology and Research

Objectives

- To get understanding of decision making in ‘C’ Language.
- To develop programming skills using different types of if Conditions.
- To impart the knowledge of switch statement in C.
- To understand the use of if…else instead of conditional operator.

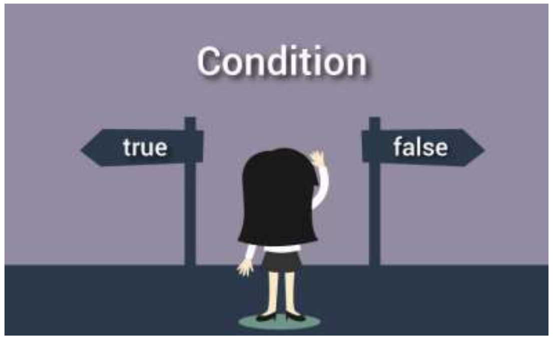
Decision Making & Branching

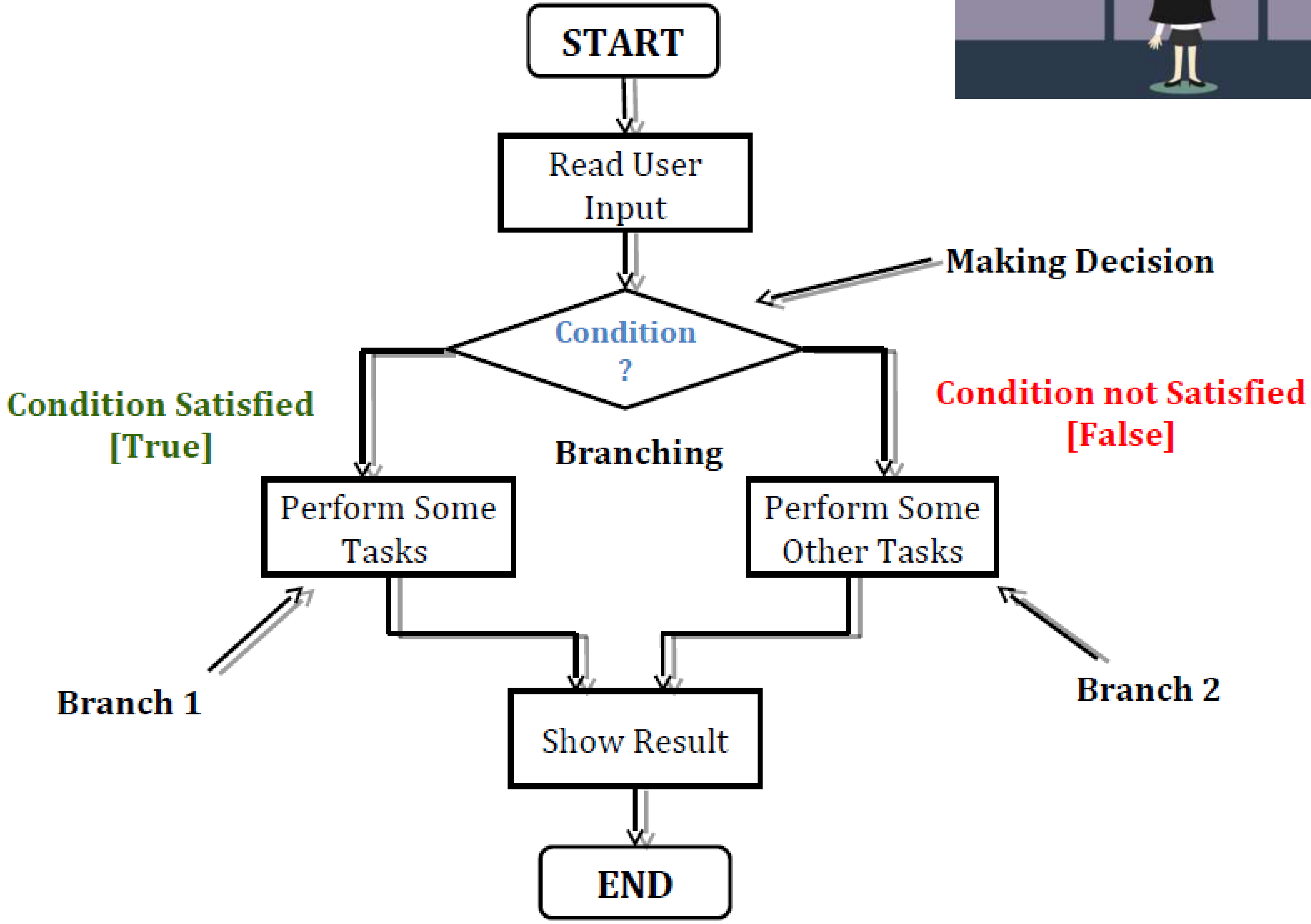
Decision making in C



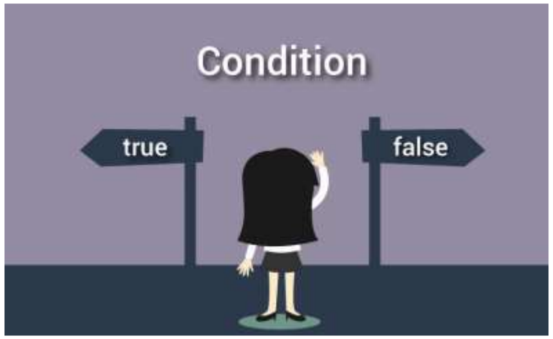
- **"Decision making and branching"** is one of the most important concepts of computer programming.
- Programs should be able to make logical (true/false) decisions based on the condition provided.
- Every program has one or few problems to solve. In order to solve those particular problems important decisions have to be made depending on the nature of the problems.
- So controlling the execution of statements based on certain condition or decision is called decision making and branching.

Decision making in C



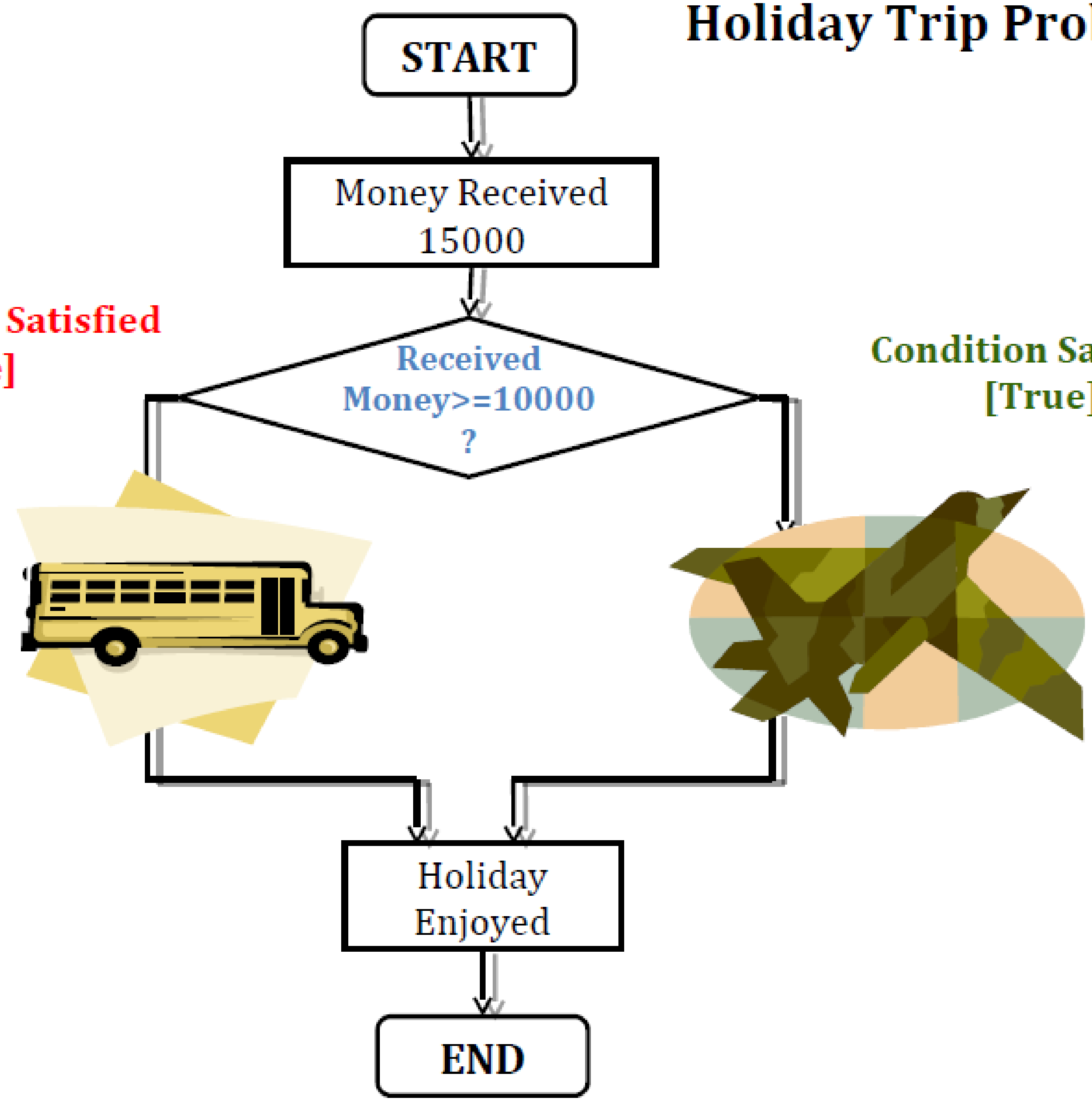


Holiday trip Problem

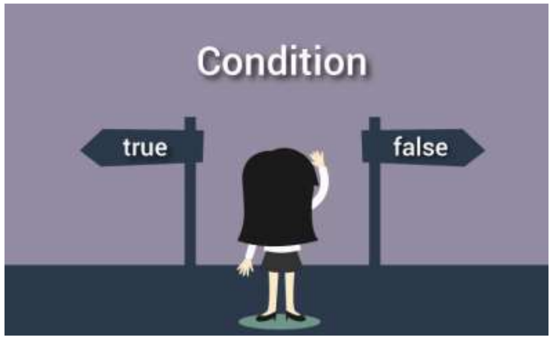


- Consider the fact that you and some of your friends have planed to go out for a holiday trip after the Semester.
- You have also decided that if you have got received money 10,000 Rupees or more from your parent then you will go out for a foreign trip.
- Otherwise, if the allotted money is less than 10,000 then you will go out for a country side trip.
- Now you are supposed to design a program to solve this problem.

Holiday Trip Problem



Decision making in C



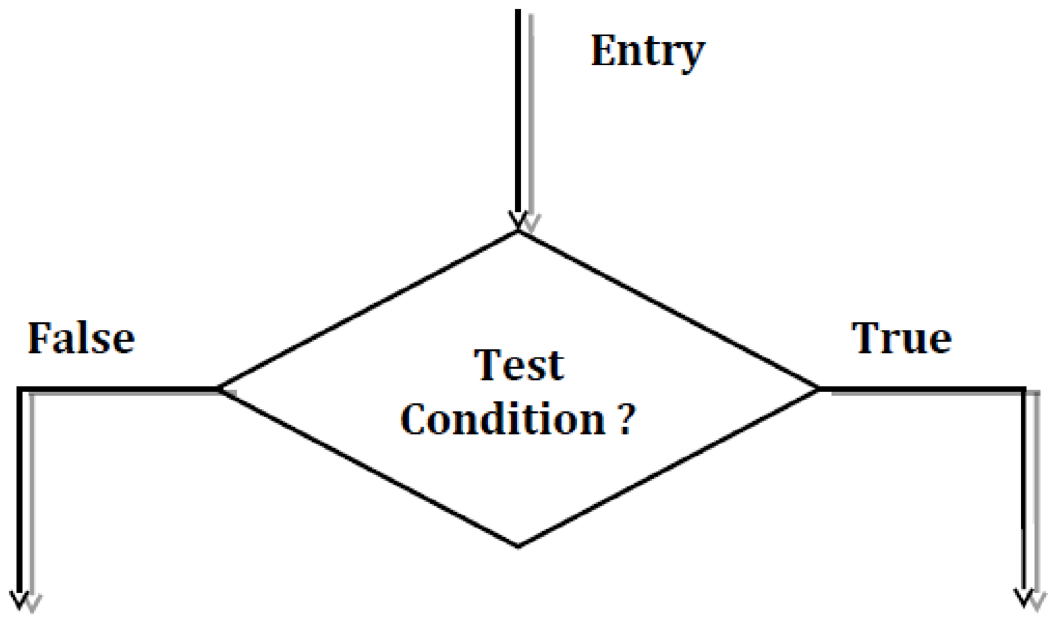
- C language possesses decision making and branching capabilities by supporting the following statements:
  - If statement
  - Switch statement
  - Conditional operator statement
  - goto statement
- These statements are knows as decision making statements.
- They are also called control statements as the control the flow of execution.

Decision making using if

- The if statement is a powerful statement for decision making and is used to control the flow of execution of statements.
- It is basically a two-way decision making statement and is used in conjunction with an expression.
- It takes the following structure:

  if (test-condition)
- It allows the computer to evaluate the expression first and then depending on whether the value of the expression or condition is true or false, it transfer the control to a particular statement.
- This point of program has two paths to follow, one for the true condition and the other for the false condition.

Decision making using if



Example:
If(Pocket balance is zero)
        Borrow money;

Decision making using if

- The if statement can be implemented if four different forms depending on the complexity of the conditions to be tested.
- The four forms are:
    - Simple if statement
    - If else statement
    - Nested if else statement
    - Else if ladder

Simple if statement

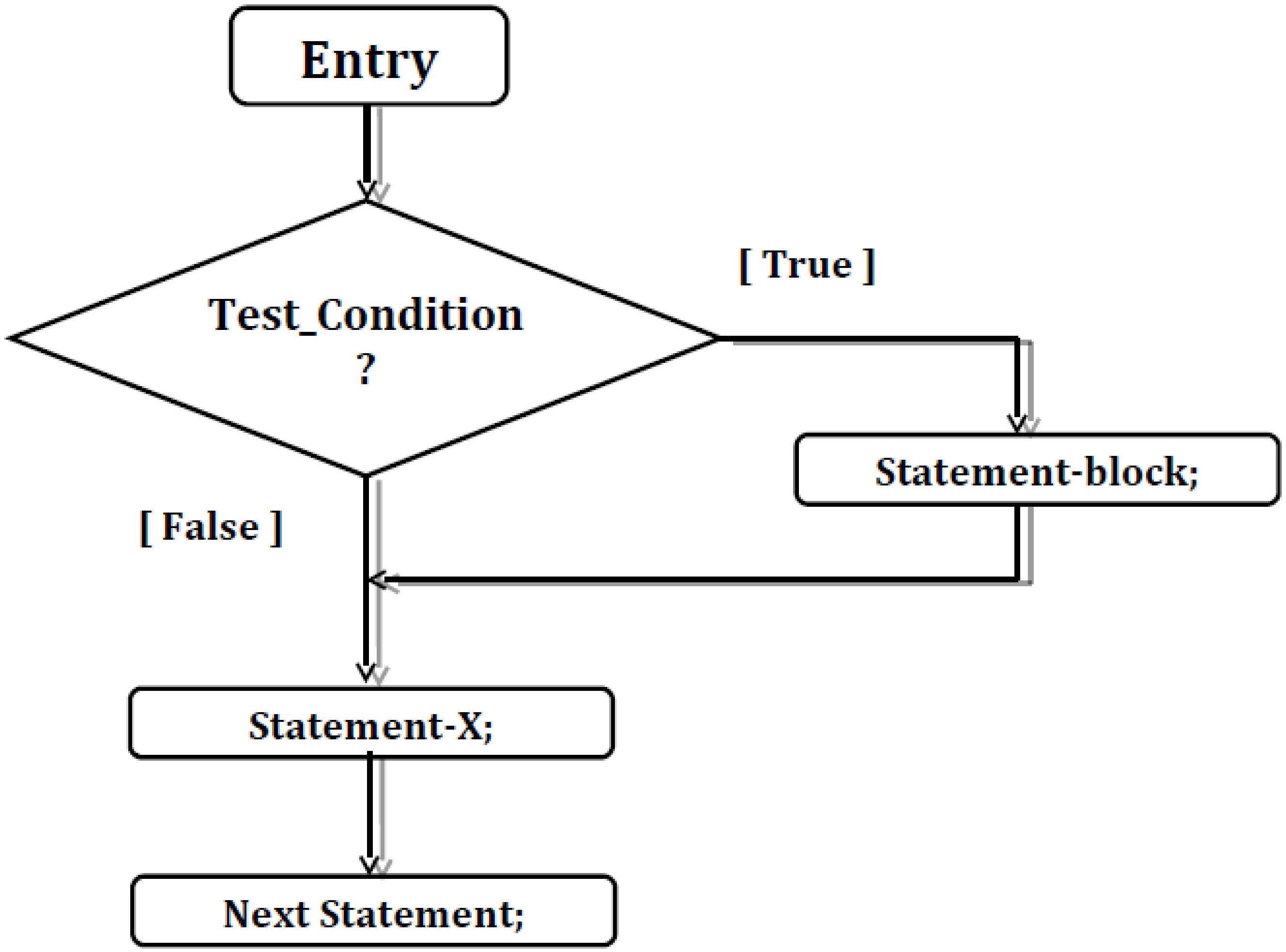
- The general form of a simple if statement is:

```
if (test_condition)
{
   statement-block;
}
statement x;
```

Simple if statement

- The general form of a simple if statement is:
- The statement block may consists of a single statement or a group of statements.
- If the test_condition is satisfied then the statement block will be executed first then the statement-x will be executed after completing the execution of statement block.
- Otherwise if the test_condition doesn't satisfied then, the statement block will be skipped and the execution will jump to the statement-x.
- So in short when the condition is true then both the statement block and the statement-x are executed but in sequence.

Simple if statement - Flowchart



Simple if statement - Example

```
#include<stdio.h>

void main()
{
  int number = 0;
  printf("\nEnter an integer between 1 and 10: ");
  scanf("%d",&number);

  if (number > 7)
    printf("You entered %d which is greater than 7\n", number);

  if (number < 3)
    printf("You entered %d which is less than 3\n", number);
}
```

if…else statement

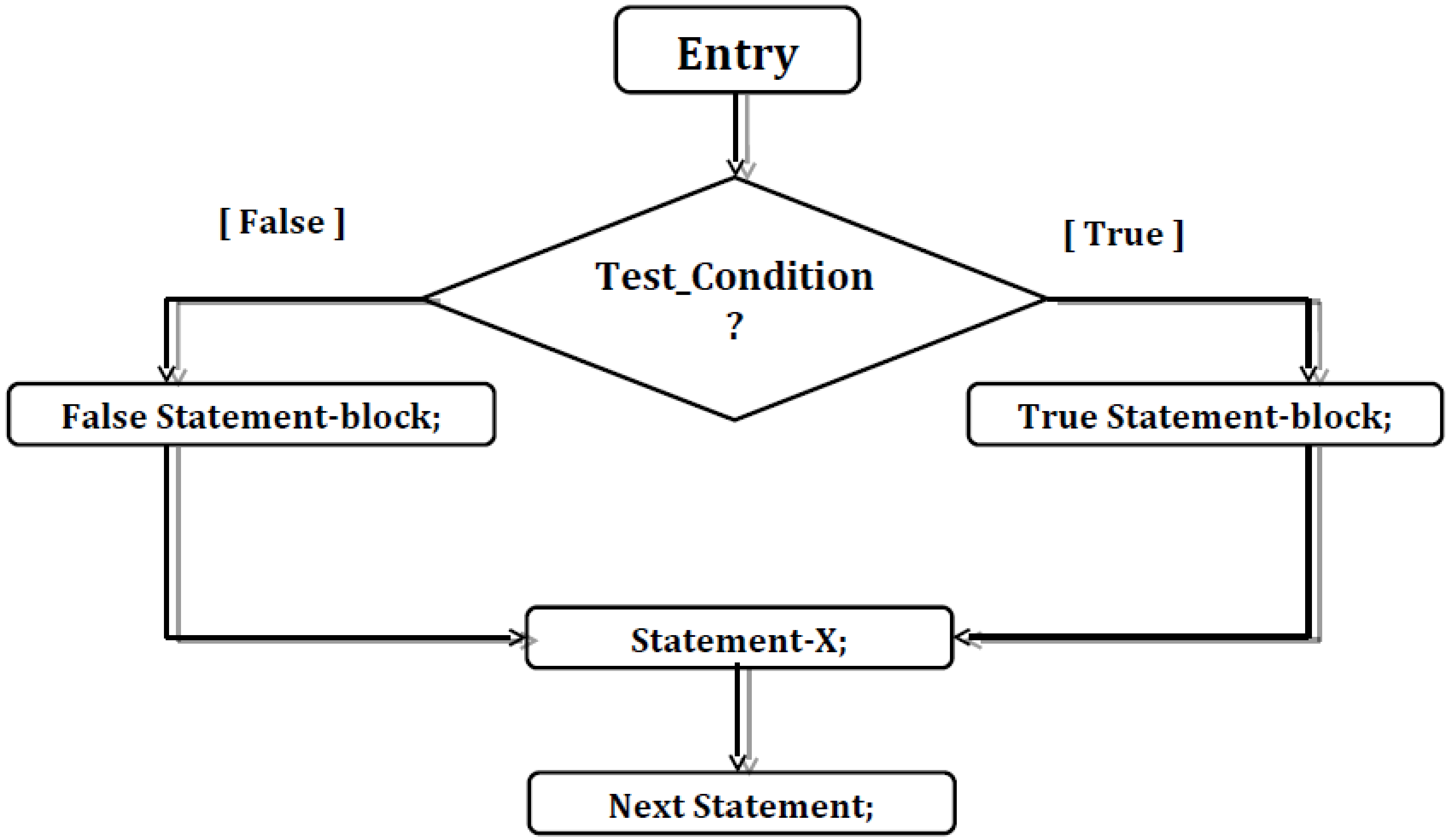
- If…else is an extension of the simple if statement.
- If the test condition is true then the true block statements, immediately following the if statements are executed.
- Otherwise the false block statements are executed.
- In short either true-block or false-block of statements will be executed, not both.
- But in both cases the control is transferred subsequently to the statement-x as it is an independent (not controlled by the if else statement) statement.
- **It is also called two way conditional branching.**

if…else statement - Structure

- The if else statement is an extension of the simple if statement.
- **The general form is :**

```
if (test_condition)
{
  True block statements;
}
else
{
  False block statements;
}
statement-x;
```

if...else statement- Flowchart



if...else statement- Example

- Example with block of statement:

```
if (marks>=40)
   {
          marks=marks+ bonus_marks;
   grade="passed";
   }
else
   {
   marks=marks;
          grade="failed";
   }
printf("The mark achieved:marks" , %d);
```

True block statement

False block statement

Nesting of if...else

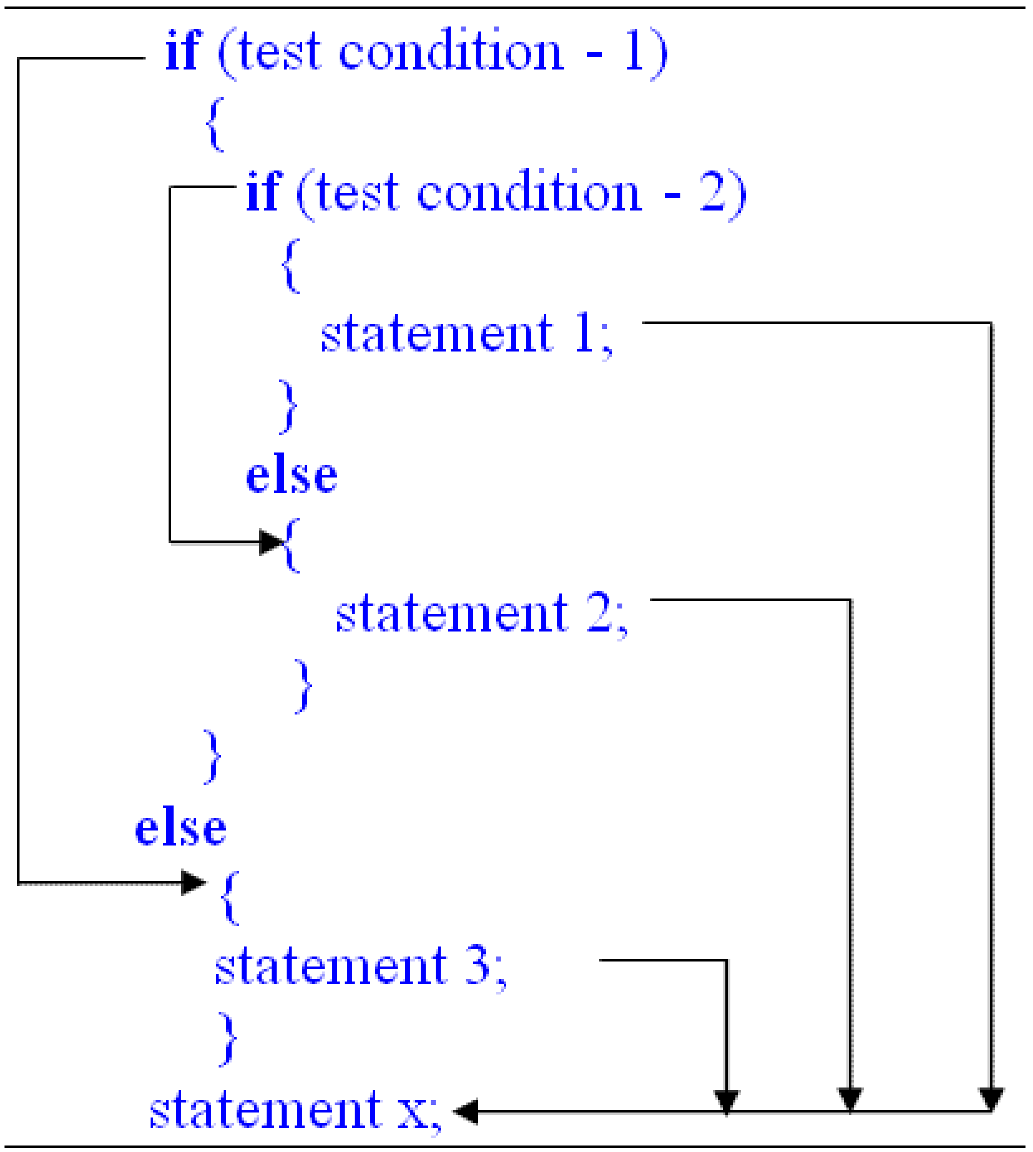
- If the series of decisions are involved, we may have to use more than one if…else statement in nested form.
- Using “if…else statement” within another “if…else statement” is called ‘nested if statement’.
- “Nested if statements” is mainly used to test multiple conditions.
- **It is also called nested conditional branching.**

Nesting of if...else - Structure

```
if (test_condition)
    {
        if (test_condition)
            {
                statement-block;
            }
        else
            {
                statement-block;
            }
    }
else
    {
     statement-block;
    }
statement-x;
```

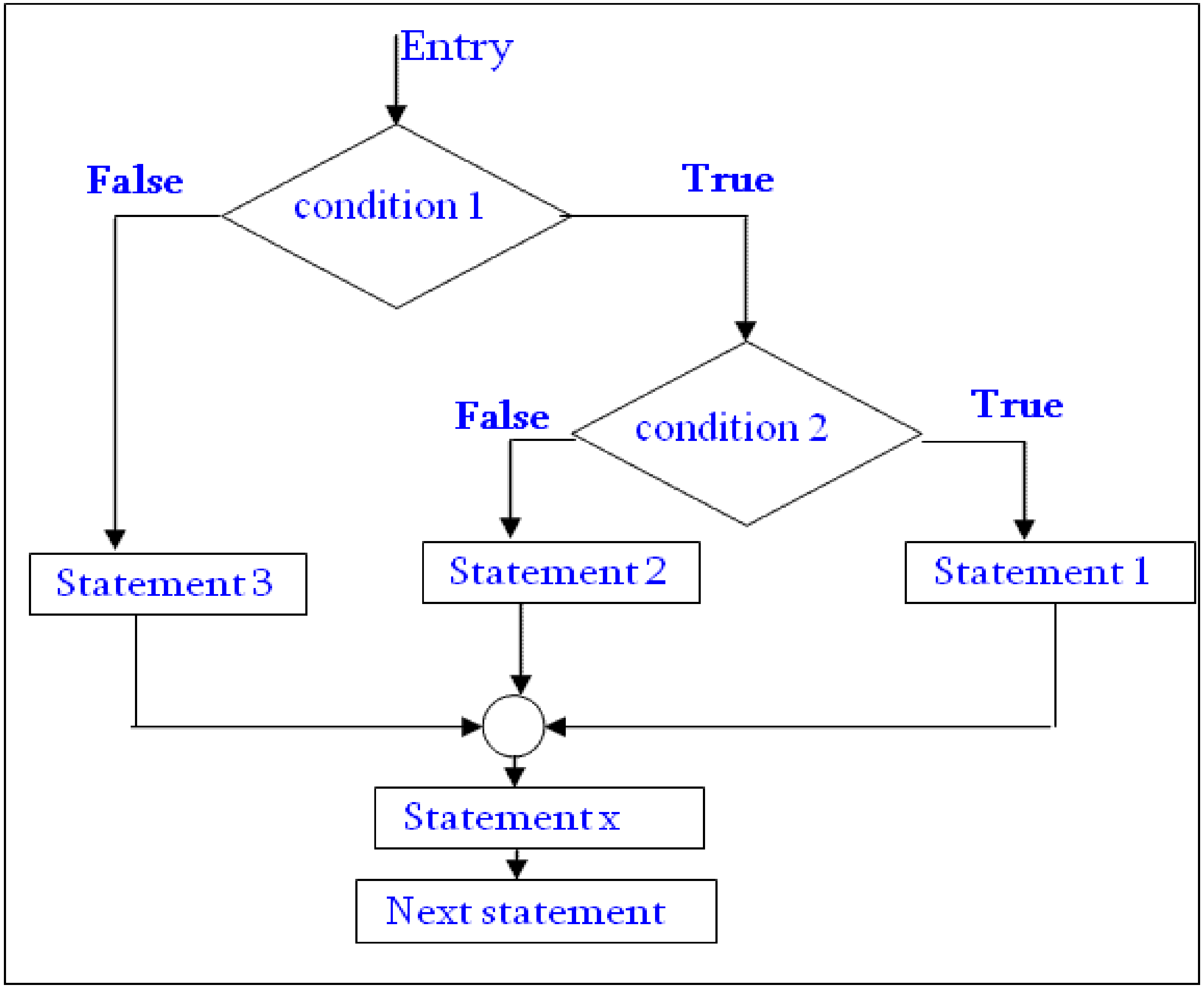
Nested if else

Nesting of if…else – Execution Flow



```
if (test condition - 1)
  {
    if (test condition - 2)
      {
        statement 1;
      }
    else
      {
        statement 2;
      }
  }
else
  {
  statement 3;
  }
statement x;
```

Nesting of if...else - Flowchart



Nesting of if...else - Example

```
if (gender==female)
{
        if (age<10)
        {
                provide free entry;
                provide free food;
        }
        else
        {
                provide only free entry;
        }
}
else
{
                statement-block;
}
statement-x;
```

$\left.\right\}$ **Nested if else**

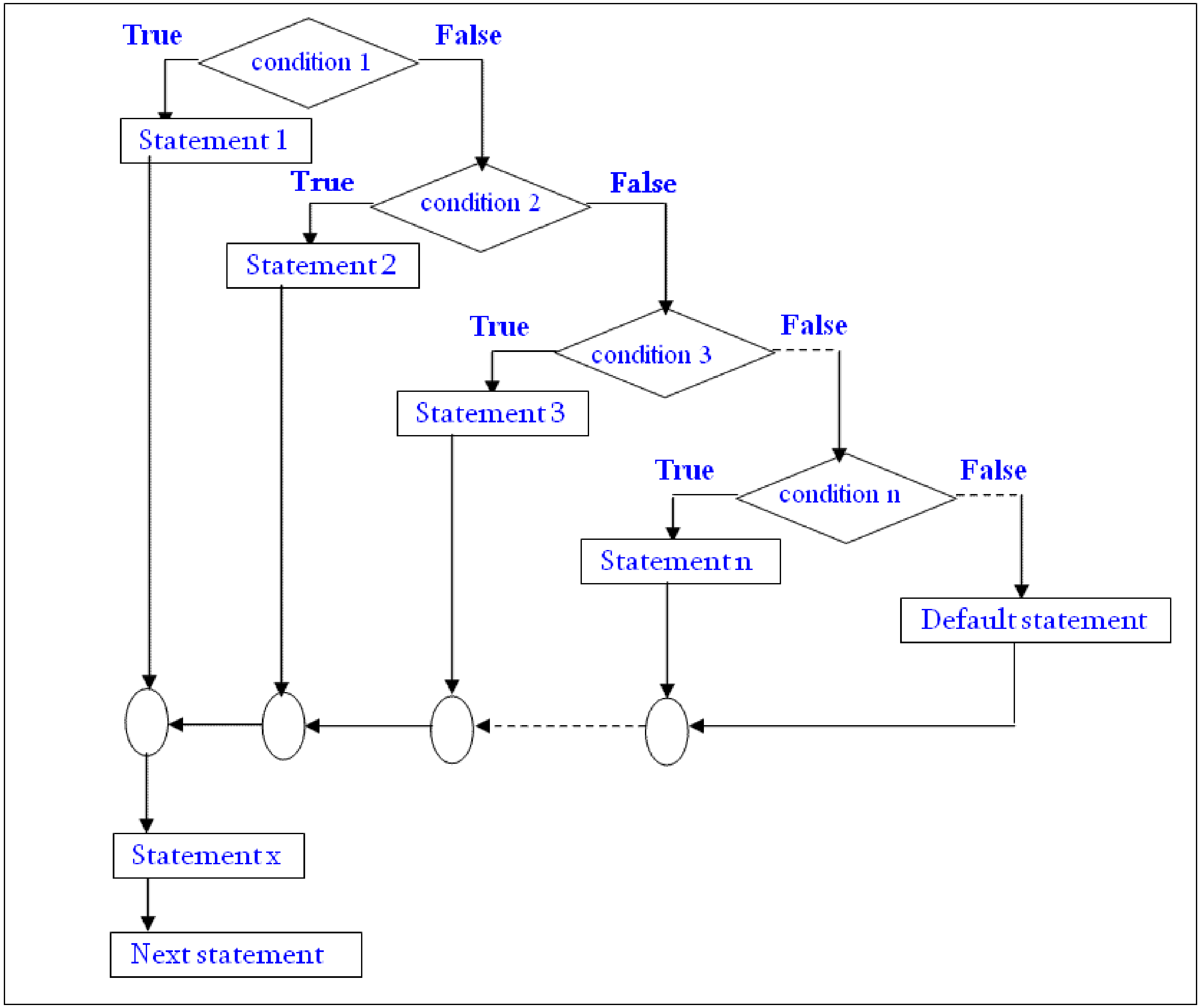
else…if Ladder

- The word ladder means the staircase.
- As the name implies this statement is used to choose right way/paths among multiple paths.
- There is another way of putting if conditions together when multiway decisions are involved.
- A multiway decision is a chain of if conditions in which the statement associated with an else condition behaves like another if condition.
- Else if ladder is also called **3 way** or **multiway** decision making statement.

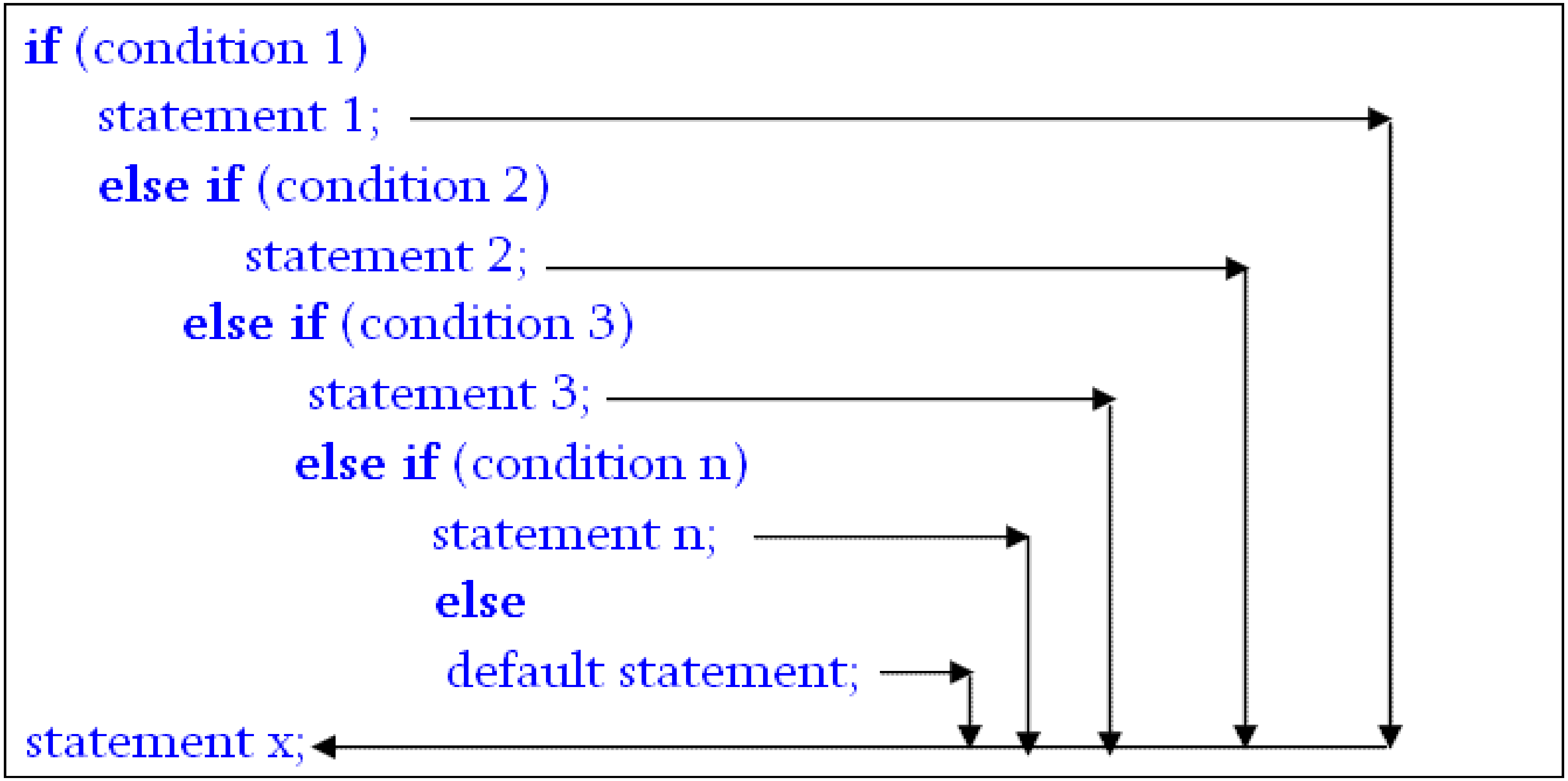
else…if Ladder - Structure

```
if (test_condition 1)
                    statement-1;
else if (test_condition 2)
                    statement-2;
else if (test_condition 3)
                    statement-3;
else if (test_condition 4)
                    statement-4;
……………………………………...
else if (test_condition n)
                    statement-n;
statement-x;
```

else…if Ladder - Flowchart



else…if Ladder – Execution Flow



```
if (condition 1)
    statement 1;
    else if (condition 2)
            statement 2;
        else if (condition 3)
                statement 3;
            else if (condition n)
                    statement n;
                    else
                     default statement;
statement x;
```

else…if Ladder - Example

```
if(mark>=50 && mark<60)
{
        printf("Your grade is D");
}
else if(mark>=60 && mark<70)
{
        printf("Your grade is C n");
}
else if(mark>=70 && mark<80)
{
        printf("Your grade is B n");
}
else if(mark>=80 && mark<90)
{
        printf("Your grade is A n");
}
else
        printf("you have failed");
```

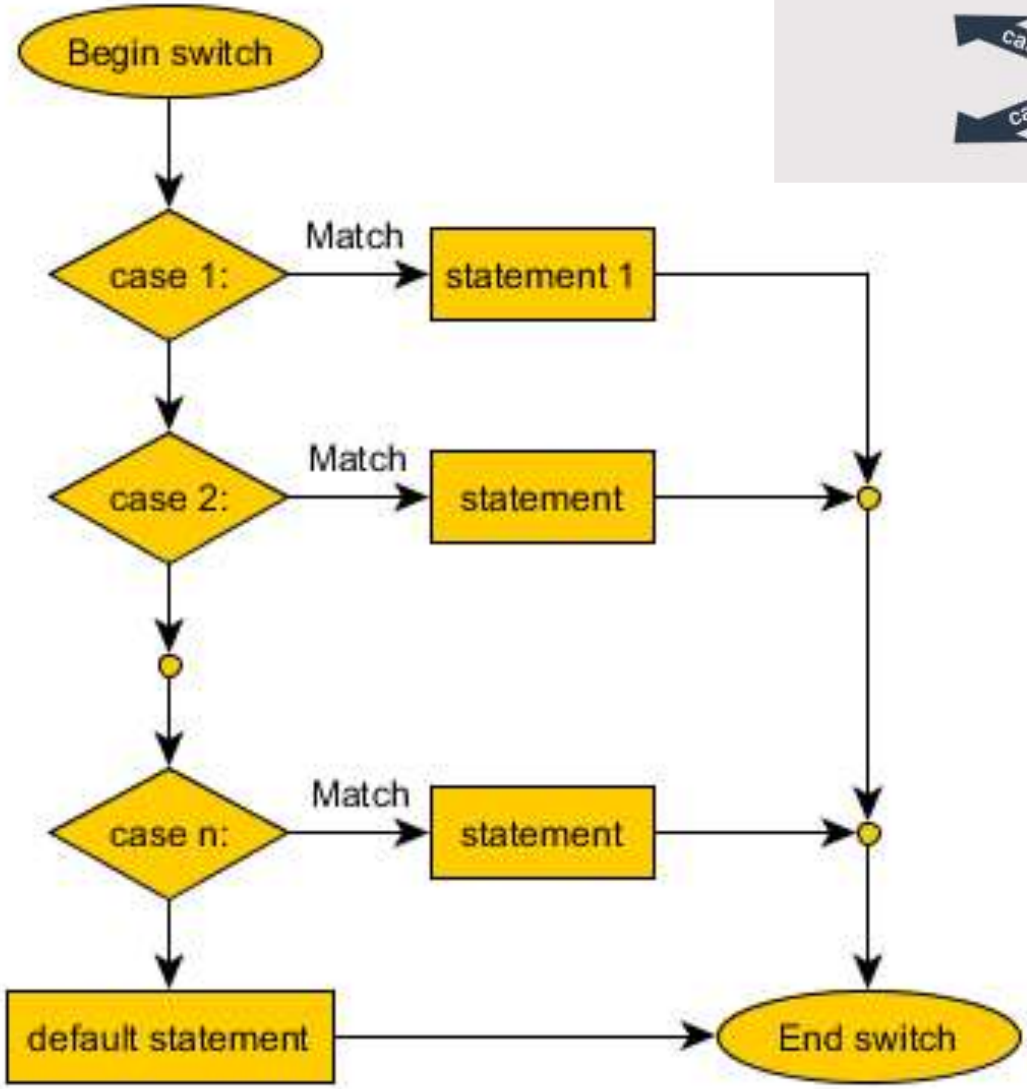
Switch Statement

- When one of the many statements is to be selected, then if conditional statement can be used to control the selection.
- However the complexity of such a program increases dramatically when the number of statements increases.
- Fortunately, C has a built in multiway decision making statement known as switch.
- The switch statement tests the value of a given variable or expression against a list of case values and when a match is found only then a block of statements associated with that case is executed.

Switch - Structure

```
switch(expression/ value)
{
        case value-1:
                        statement-block-1;
                        break;
        case value-2:
                        statement-block-2;
                        break;
                        ..........................
        case value-n:
                        statement-block-n;
                        break;
        default:
             default-statement-block;
             break;
}
statement-x;
```

Switch - Flowchart



Switch - Example

```
#include<stdio.h>

void main( )
{
   int a, b, c, choice;

   while(choice != 3)
   {
      /* Printing the available options */

      printf("\n 1. Press 1 for addition");
      printf("\n 2. Press 2 for subtraction");
      printf("\n Enter your choice");

      /* Taking users input */
      scanf("%d", &choice);
      switch(choice)
       {
         case 1:
            printf("Enter 2 numbers");
            scanf("%d%d", &a, &b);
            c = a + b;
            printf("%d", c);
            break;
         case 2:
            printf("Enter 2 numbers");
            scanf("%d%d", &a, &b);
            c = a - b;
            printf("%d", c);
            break;
         default:
       printf(“wrong key");
       printf("\npress any key to continue" );
       }
   }
}
```

Rules for Switch Statement

- The switch statement must be an integral type.
- Case labels must be constant or constant expression.
- Case labels must be unique. No two labels can have the same value.
- Case labels must end with colon.
- The break statement transfer the control out of the switch statement.
- The break statement is optional. So two or more case labels may belong to the same statements.
- The default label is optional. If present, it will be executed when the expression does not find a matching case label.
- There can be at most one default label.
- The default may be placed any where but usually placed at the end.
- It is permitted to nest switch statements.

- **Valid expressions for switch:**

```
// Constant expressions allowed
        switch(1+2+23)
        switch(1*2+3%4)
```

- **Invalid switch expressions for switch:**

```
// Variable expression not allowed
        switch(ab+cd)
        switch(a+b+c)
```

Conditional Operator

- The C language has an unusual operator which is useful for making two way decisions.
- This operator is a combination of ? and :
- It takes three operands. This operator is popularly known as the conditional operator.
- The conditional operator can be used as the replacement of if else conditional statement for two way decision making.

Conditional Operator

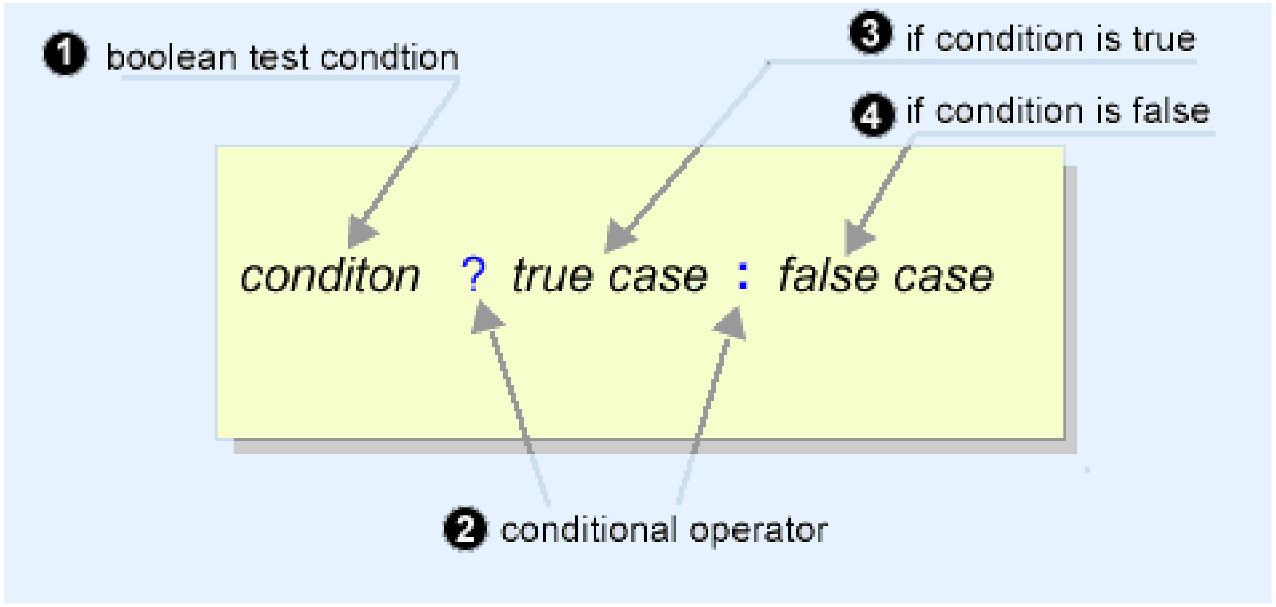
- The general structure of conditional operator:

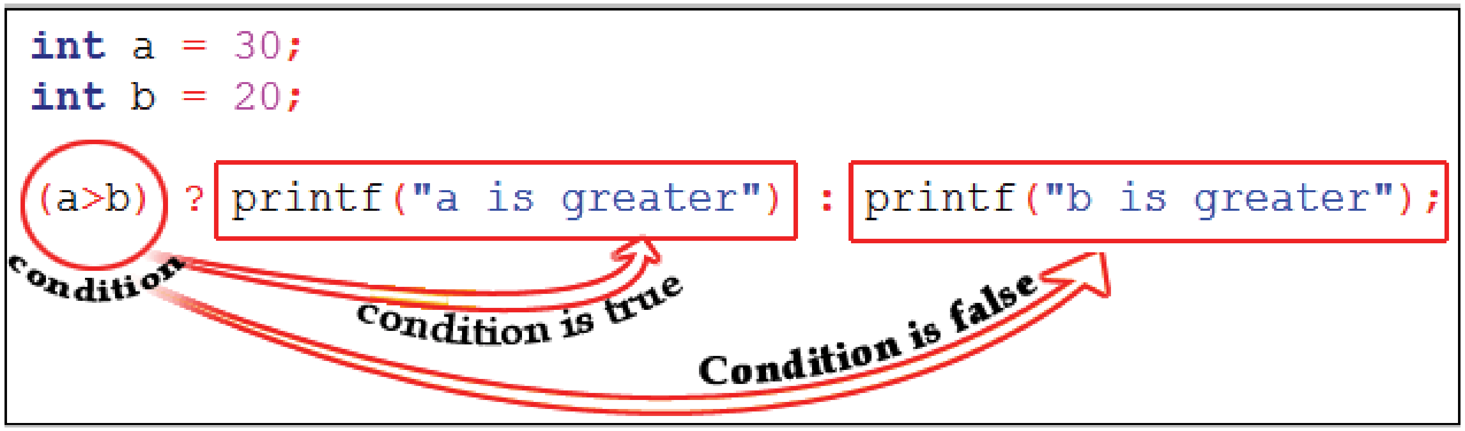
  Conditional expression? true-statement 1: false-statement;

- The condition is evaluated first. If the result is true then the statement 1 is executed and its value is returned.
- Otherwise statement 2 is executed and its value is returned.
- Example:

  flag = (x<0) ? 0 :1;

Conditional Operator - Execution





```
int a = 30;
int b = 20;
(a>b) ? printf("a is greater") : printf("b is greater");
```

Use of if…else instead of conditional operator

- Conditional operator:

```
flag = (x<0) ? 0 :1;
```

- It is equivalent to:

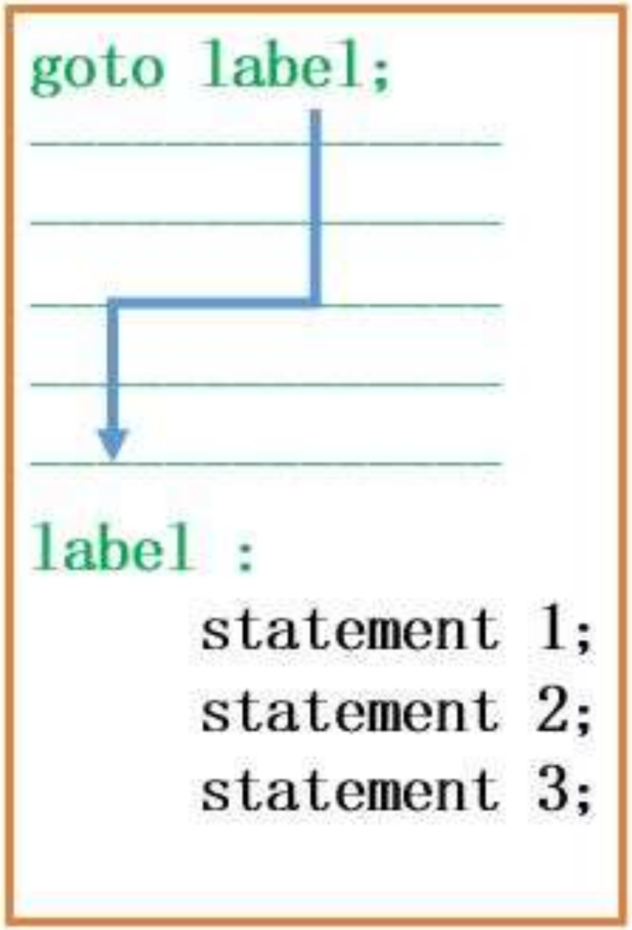
```
if(x<0)
    flag=0;
else
    flag=1;
```

goto statement

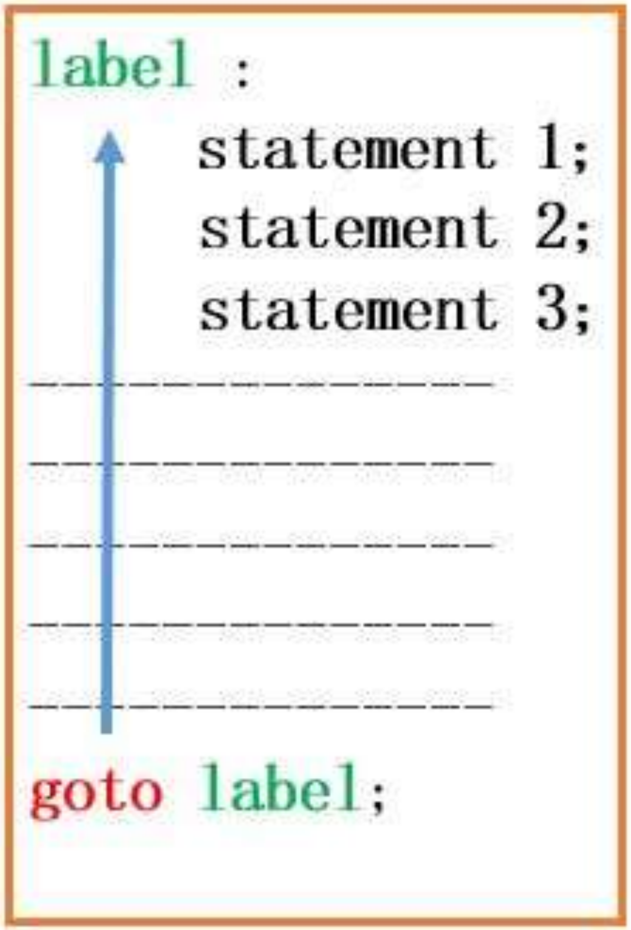
- A goto statement in C programming provides an unconditional jump from the 'goto' to a labeled statement in the same function.

```
Syntax1         |  Syntax2
----------------------------
goto label;     |   label:
.               |   .
.               |   .
.               |   .
label:          |   goto label;
```

goto statement



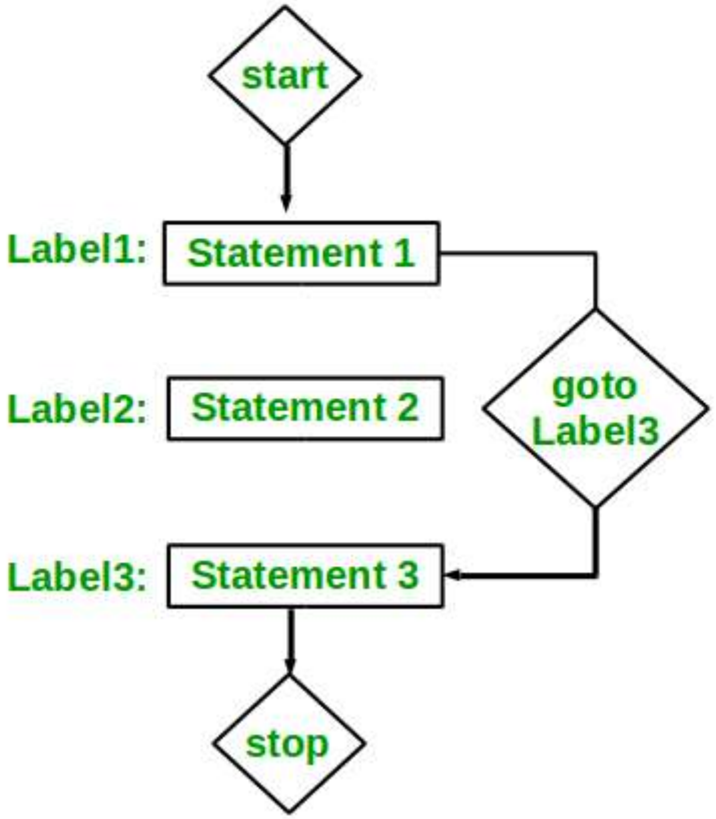
Forward Reference



Backward Reference

goto statement - Example

```
#include <stdio.h>
int main()
{
  int sum=0;
  for(int i = 0; i<=10; i++)
     {
             sum = sum+i;
             if(i==5)
             {
               goto addition;
             }
         }
         addition:
         printf("%d", sum);
         return 0;
     }
```



Disadvantages of using goto statement

- The use of goto statement is highly discouraged as it makes the program logic very complex.
- Use of goto makes really hard to modify the program.
- Use of goto can be simply avoided using break and continue statements.

Previous Year Questions

1. Which are the keywords in switch statement?
2. Differentiate: nesting of if vs else….if ladder statements.
3. Write a program to make simple calculator using arithmetic operators as a input using switch…case.
4. Explain else..if ladder with example.
5. Write a program to implement user iterative calculator. The program allows four operation +,-,*,/. For / operation, also print an error if division is not possible.
6. Explain nested if and switch statement with Example.
7. Replace code using if…else statement.

```
que = (q1>q2)?((q1>q3)?q1:q3):q2;
```